2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) | 978-1-6654-4899-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/CVPRW53098.2021.00013

From Evaluation to Verification: Towards Task-oriented Relevance Metrics for Pedestrian Detection in Safety-critical Domains

Maria Lyssenko*†, Christoph Gladisch*, Christian Heinzemann*, Matthias Woehrle*, Rudolph Triebel†‡
*Robert Bosch GmbH, Corporate Research, Robert Bosch Campus 1, 71272 Renningen
firstname.lastname@de.bosch.com
†Technical University of Munich, Boltzmannstrasse 3, 85748 Garching
firstname.lastname@in.tum.de
‡German Aerospace Center (DLR), Münchener Strasse 20, 82234 Wessling
rudolph.triebel@dlr.de

Abstract

Whenever a visual perception system is employed in safety-critical applications such as automated driving, a thorough, task-oriented experimental evaluation is necessary to guarantee safe system behavior. While most standard evaluation methods in computer vision provide a good comparability on benchmarks, they tend to fall short on assessing the system performance that is actually relevant for the given task. In our work, we consider pedestrian detection as a highly relevant perception task, and we argue that standard measures such as Intersection over Union (IoU) give insufficient results, mainly because they are insensitive to important physical cues including distance, speed, and direction of motion. Therefore, we investigate so-called relevance metrics, where specific domain knowledge is exploited to obtain a task-oriented performance measure focusing on distance in this initial work. Our experimental setup is based on the CARLA simulator and allows a controlled evaluation of the impact of that domain knowledge. Our first results indicate a linear decrease of the IoU related to the pedestrians' distance, leading to the proposal of a first relevance metric that is also conditioned on the distance.

1. Introduction

Automated driving applications impose high demands on the safety of perception functions [4] as an automated vehicle requires precise knowledge of its surroundings for behaving safely. As one particular example, consider a camera-based pedestrian detection function in an automated vehicle that shall recognize and localize pedestrians in camera images. Ideally, such a function should detect every pedestrian that is contained in an image as a non-detected pedestrian may lead to a safety-critical situation. Obviously, not every missed pedestrian leads to a safety-critical situation. It depends on the (physical) state of the traffic situation if a misdetection may indeed lead to a safety concern.

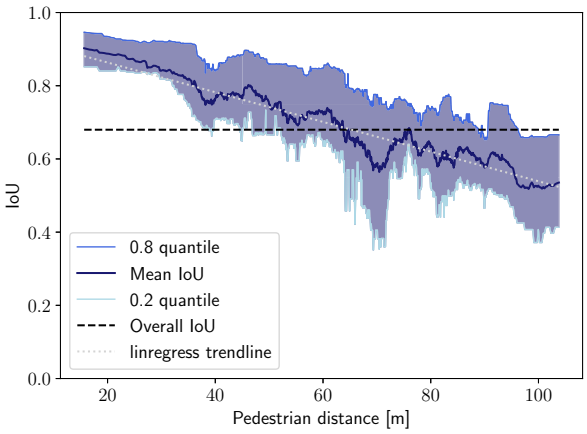


Figure 1: Relationship of a standard computer vision performance measure (IoU) for a pedestrian detection task with instance-wise pedestrian distances based on test data acquired from CARLA. The results show a linear trend for decrease in performance over distance, leading to the observation that IoU alone is not sufficient for arguing safety.

Even though misdetections might be evaluated quite differently depending on the traffic situation from a safety perspective, there is yet an "insufficient consideration of safety in metrics" [20]. This is because the usual computer vision metrics for rating the performance of a (deep learning-based) detector or segmentation algorithm like mean average performance, intersection over union (IoU), etc. are generic and, thus, task-agnostic. As an example, metrics

978-1-6654-4899-4/21/$31.00 ©2021 IEEE
DOI 10.1109/CVPRW53098.2021.00013

for semantic segmentation treat all classified pixels equally. As a consequence, incorporating a notion of safety into the performance evaluation of a perception function requires a more detailed analysis of the predictions.

On the level of behavior of automated vehicles, domain-specific threat metrics [7] have been introduced for rating the criticality of a traffic situation for the planned future behavior of a vehicle. These encode knowledge about physics and possibilities for future behavior for assessing the situation at hand. As previous works have suggested [11, 20, 19], we believe that consideration of domain knowledge and the use of task and domain-specific metrics are necessary for evaluating the performance of deep learning-based computer vision functions for safety-critical tasks. As for all production systems, metrics should reflect that the system performs acceptably in all relevant data slices [3]. We can see such a refinement of metrics as a better specification of our system, *i.e*. we want to avoid underspecification [8]. In our example, it might be permissible to miss some far-away pedestrians, but the detector needs to be perfect for the pedestrians that matter from a safety perspective. While our focus is on better specification and verification of system performance, previous work has used a similar approach for identifying corner cases [2], *i.e*. for falsification.

In this paper, we present initial work towards incorporating task-oriented domain-knowledge into the assessment of the detection quality of a deep learning-based perception function. As a first step, we use the distance of other traffic participants as a proxy metric for the use of more elaborate threat metrics, as these threat metrics also rely on such information [7]. To this end, and as our second contribution, we present an experimental setup based on the open-source simulator CARLA [9] that provides the opportunity to perform structured and controlled experiments for evaluating the impact of domain knowledge. In particular, we show how the acquisition of ground truth data for our analyses can be compiled by fusing and post-processing the information provided by CARLA to assist other researchers.

We evaluate our approach on DeepLabv3+ network for semantic segmentation [5]. We train and evaluate with data from the CARLA simulator. In particular, we investigate the impact of distance of pedestrians on the segmentation quality. Based on this, we derive a first proposal for a task-oriented metric that incorporates distance as a proxy for relevance of pedestrians. Our evaluation results indicate that (1) performance of pedestrian detection drops linearly with distance as shown in Figure 1 and (2) for our test set we were able to find a threshold for the detection condition such that every pedestrian within a given distance was detected.

The remainder of this paper is structured as follows. We present related works in Sec. 2. Sec. 3 introduces our CARLA setup and Sec. 4 our data acquisition approach based on our CARLA setup. We present our experimental results in Sec. 5 before concluding the paper in Sec. 6.

2. Related Work

In the following, we discuss related works introducing a notion of safety-motivated relevance for the assessment of the prediction quality of neural networks. In the terminology of Sämann et al. [16], such metrics try to bridge the gap between traditional prediction quality metrics and task-oriented safety metrics.

As in our approach, distance of objects to the vehicle is often used as representative for relevance, where lower distance is often used synonymously to higher relevance which requires higher detection rates [11]. Bolte et al. [2] present a corner case detection system where they can also filter based on relevance. As in our work, they consider moving objects nearer to the vehicle as more relevant, but they use the distance of the bounding box from the bottom line of the image as a proxy metric for a real, measured distance. While we follow a simulation-based approach, distances in real data may also be obtained based on measurements with a reference sensor like Lidar as performed in the A2D2 dataset [12]. We note though that our approach aims at a single detection framework that is useful for all distances, and not a specialized method only for certain distances [18].

Philion et al. [15] define relevance from a planning-perspective, *i.e*., a misdetection is relevant if the vehicle's behavior planner performs a different action based on ground truth detection compared to the actual detection. In contrast to our approach, this is an end-2-end metric that can only be evaluated for an entire AV system rather than a single component.

Volk et al. [19] define a combined safety metric for evaluation perception systems that includes a safety component based on the responsibility sensitive safety (RSS, [17]) approach. In contrast to our approach, their detection quality assessment relies on object tracking metrics evaluated over consecutive frames, while we focus on single images.

3. Pedestrian-oriented Simulation Setup

In this section we describe our methodology to exploiting synthetically generated pedestrian data from CARLA to introduce an task-oriented relevance metric. In our Experiment Setup in Section 5.1 we test a DNN-based (Deep Neural Network) computer vision function that performs semantic segmentation as our system under test (SUT). This task can be seen as object classification on pixel-level, mapping each pixel from the RGB image to a predefined class ID leveraging semantic class definitions in the style of the Cityscapes dataset [6]. Therefore, we lay out simulation environment with focus on pedestrians, followed by bringing the 3D world actors to the pixel grid.

3.1. CARLA Software Environment

CARLA is as an open-source simulator targeting the development and validation of autonomous driving systems. The platform provides a variety of digital assets including pedestrians with diverse appearances and several vehicle models, urban 3D maps, light-effects, and sensors, complemented by a wide blueprint library implementing dynamic behavior. With a flexible API pedestrians and vehicle actors can be spawned within the CARLA simulator framework in a controlled way, setting initial location, waypoints, velocity, and orientation. We leverage this simulation platform as it provides a controlled scene creation and image generation with precise ground truth at a very low cost. However, to be useful for our purpose, we needed to leverage and wrap the CARLA API into an own framework as described next.

3.2. Controlled Actor Spawning

Our goal is to evaluate computer vision-based perception functions, particularly targeting the detection of pedestrians. We claim that for this purpose we need to cover a variety of visual traffic scenarios within the operational design domain (ODD). Based on CARLA, we created a framework that allows us to plugin diverse combinations of scene generators for the corresponding evaluation purposes. In this work we focus on the development of an actor spawning module, giving control over pedestrians, vehicles, and camera positions. To focus on these parameters only we keep the weather conditions constant. As a result, this setup enables us to identify the effects of physical object properties on the performance of our investigated perception function (see Section 5).

In our setting we solely populated actors using CARLAs pre-built map *Town03*, where the camera was attached to an autopilot vehicle progressing at a constant speed. Since the number of available spawning locations for vehicles is limited to 265 road points of our map *Town03*, we placed this precise quantity of vehicles inside our simulation using listed models from the blueprint library.

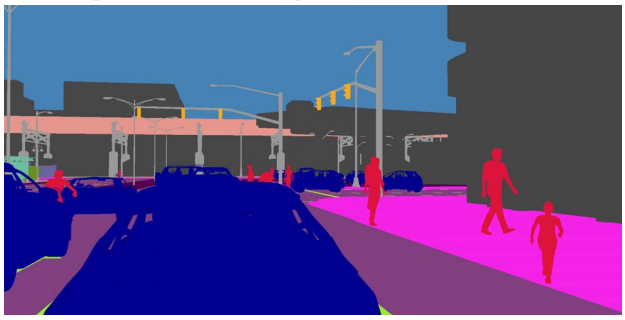
3.3. Making the Pedestrians Move

To identify the influence of physical properties on the object perception, given a predefined distance and hip joint orientation, a controlled and accurate pedestrian placing in reach of the ego vehicle is implemented. With the creation of realistic traffic scenarios in mind, we focus primarily on spawning the pedestrians on the sidewalk. We also contribute occasional waypoints from the road or shoulder lane type to introduce critical scenarios in our scene creation.

Originally, the spawned pedestrians show static characteristics only. To add dynamic behavior, we attach an AI-controller (WalkerAIController) to each pedestrian actor. The aforementioned interface from the blueprint library assigns a specific lane type destination, making the walkers

(a) CARLA RGB output

(b) Depth frame containing pixel-wise distance information



(c) Corresponding ground truth with 23 labeled classes

Figure 2: A CARLA traffic scenario with pedestrians placed on specific lane types (sidewalk, crosswalk, shoulder) at different locations and with random hip joint rotations (see Section 3.1). Figure a depicts an RGB frame rendered with the Unreal Engine, while Figure b and Figure c show the corresponding depth and semantic map.

wander around at random speed, thereby enabling different body postures as shown in Figure 2a. Inside the simulation, all actors are associated with a unique *identification* containing *attributes* from the blueprint library supplemented by the actors *transformation*, *velocity* and *acceleartion*.

The current CARLA version 0.9.11 provides a semantic group segmentation sensor. However, this neither includes an instance segmentation nor an object detection sensor, which we need in our evaluation to separate actors in the image domain. This constraint poses a significant challenge during data retrieval, namely: How can we distinguish be-

tween actor pixels in the image domain that correspond to the same class? This will be detailed in the following section.

Algorithm 1 Bounding box (BB) retrieval and pedestrian's instance-wise depth assessment

Input: D_T, A_i, p_{max}
$D_T \leftarrow$ unfiltered test dataset, **not** $\emptyset$
$A_i \leftarrow$ number of available actors p in sample i
$p_{\text{max}} \leftarrow$ upper threshold for each sample $i \in D_T$
$P_T := \{\, i \in D_T \mid 1 \leq A_i \leq p_{\text{max}} \,\}$
Output: Dataframe (D_f) containing all valid p instances

for $i \in P_T$ **do**
  for $p \in A_i$ **do**
    FORWARD PROJECTION
      Use cam_{intr} with cam_{extr} to project $\text{BB}_{\text{3D}}(p)$ on the image plane ($\text{BB}_{\text{2D}}(p)$).
    BOUNDING BOX VALIDATION
      if $\text{BB}_{\text{2D}}(p)$ lies within the image sensor **and** pixel(p) acquired from GT > 0 at this location **then**
        $\text{BB}_{\text{2D}}(p)$ is valid = **True**
      end if
    INSTANCE-WISE DEPTH ASSESSMENT
      if BB_{2D} is valid **then**
        $\text{BB}_{\text{2D}}(p)$ upper-left position, width, and height to crop corresponding GT, prediction, and depth map regions.
        Binary mask extraction from GT-crop to calculate pixel-wise distance values from depth map using mean and median distance of p.
        Computer vision performance measures (IoU, Sensitivity) estimation using overlap between GT and prediction.
      else
        Continue with **next** p
      end if
    D_f update using p (mean, and median distance, IoU, Sensitivity)
  end for
  File output of D_f for correlation analysis
end for

4. Our Data Acquisition Approach

This section describes our data acquisition approach based on the CARLA setup introduced in the previous section. In particular, we describe the settings that we used (Section 4.1) and our extensions to CARLA that we needed to implement for compensating the missing instance segmentation in CARLA (Section 4.2). Finally, we provide some details on the dataset that we recorded for our evaluations (Section 4.3).

4.1. Camera Types and Settings

To obtain the image data from the surroundings we use a specific actor type from the blueprint library to stream data at each simulation step. Our first camera is the RGB sensor with a bit depth of 8 bit. Due to the Depth camera, raw data of our scene is acquired and stored as pixel-wise distance information to the camera, representing the z-buffer. Our third Semantic Segmentation camera performs the aforementioned semantic segmentation task classifying each pixel of non-occluded objects in the image.

All three cameras (RGB, Depth, and Semantic Segmenation) from Figure 2 are attached to our ego vehicle with a resolution of 2048×1024 and 90° horizontal field of view. The intrinsic camera parameters, including *e.g.* focal length, are obtained from CARLA. We define the extrinsic parameters (cam_{extr}), placing the camera with 1° pitch, 10° yaw, and 1.5 meters height. cam_{intr} and cam_{extr} are used to define the camera model that will be used in Section 4.2 to perform the forward projection onto the image domain.

For the calculation of our relevance metrics on pedestrian instances we require, however, instance segmentation which assesses different IDs to overlapping instances of the same semantic class, delivering distinguishable pedestrian pixels. While generating our CARLA image database, we limit ourselves to a maximum of 20 pedestrians per frame, positioned within a range of 2 to 120 meters in reference to the ego vehicle, offering hip joint rotations from 0° to 180°.

4.2. From CARLA to Image Domain

Since CARLA neither provides a specific camera sensor for instance segmentation nor object detection in the image domain, we introduce an approximation to map CARLA's 3D pedestrian bounding boxes BB_{3D} to the 2D pixel grid BB_{2D} of the RGB camera. To acquire the pixel coordinates of the pedestrian bounding boxes we use the forward projector model transformation from [10]. Applying the aforementioned camera model (cam_{intr} and cam_{extr}), followed by a validation check with the semantic ground truth, we discard invalid or entirely occluded pedestrians, as described in Algorithm 1. Since CARLA provides active actor IDs regardless being visible in the current frame we leverage semantic segmentation ground truth (GT) to affirm pedestrians pixel presence within the questioned bounding box projection before calculating computer vision performance measures like IoU, sensitivity, etc..

4.3. Dataset Details

We split the generated data into training, validation and testing samples as shown in Table 1. Note that for the testing dataset, we filtered by the number of pedestrians

per frame to alleviate crowded samples. In particular, we only considered images containing a maximum number of 7 pedestrians ($p_{\max}$, *c.f.* Algorithm 1 from Section 4.2) per frame, counting 1196 pedestrian instances in total. This operation reduced the test split from an original size of 665 images to a total of 486 images (D_T and P_T in Algorithm 1).

Train	Validation	Test
3450	500	486 (665)

Table 1: Overview of the synthetic dataset split. The samples across the splits were acquired from CARLA's *Town03* using all *265 spawning points* to record the scenery, where the change of ego vehicle's spawning point enabled a random rearrangement of pedestrians from 2 to 120 meters.

Figure 3 shows the distribution of pixel-wise distances for each dataset, which are fairly similar across the split. As outlined in Section 3.3, unfortunately, without an instance segmentation sensor, and the resulting bounding box mapping inaccuracies, we focus on group segmentation rather than creating a pedestrian count statistics. However, the instance-wise distance distribution among the dataset splits could be considered further, evaluating different pedestrian statistics in the future.

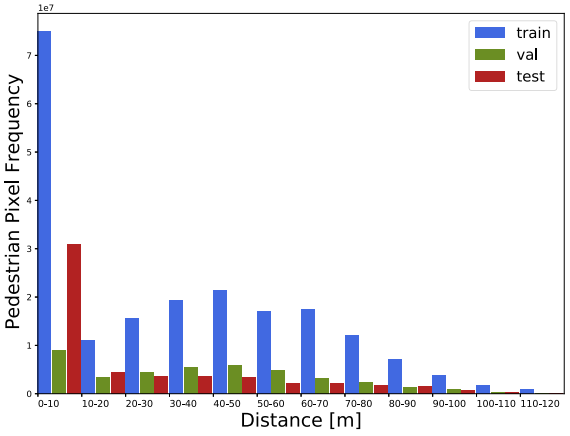


Figure 3: Distribution of pixel-wise distances in our synthetic dataset splits targeting the pedestrian class. As a result of near range pedestrians covering a larger pixel area, the histogram denotes peaks within a range of 0 to 10 meters. However, to discuss and mitigate data imbalance from pedestrian count perspective, instance segmentation is required to provide a detailed pedestrian distribution across the splits.

5. Experiments

The basis for the evaluation is the synthetic image data from Section 4.3. We perform dedicated experiments to investigate the correlation between physical attributes and widely used computer vision performance metrics. In the following, we first describe our experiment setup before showing our results.

5.1. Experiment Setup

Since we focus on effects of varying pedestrian distances, we eliminate variations on camera parameters and weather, i.e. all scenes are at day-time. [1]

Our DNN uses DeepLabv3+ [5] architecture with the ResNet-101 backbone, incorporating an output stride of 16 with batch normalization.

DeepLabv3+ is trained on image crops, *i.e.* during training we randomly sample sub-images from the 2048×1024 high resolution CARLA image to create 513×513 crops as input to the DNN.

For the implementation we use PyTorch [14] and train our DNN on the basis of mini-batch Stochastic Gradient Descent (SGD) with a momentum of 0.9, l_2 weight decay of 5e-4 and a base learning rate (lr_{base}) of 3e-4 applying a learning rate scheduler using Equation 1 from [21]

$$\text{lr} = \text{lr}_{\text{base}} \left(1 - \frac{\text{iter}_{\text{global}}}{\text{epoch}_{\text{total}} \text{iter}_{\text{num}}}\right)^{0.9} \quad (1)$$

to calculate the current learning rate (lr) for the given global step ($\text{iter}_{\text{global}}$). In total we train our DNN for 100 epochs ($\text{epoch}_{\text{total}}$) with a mini-batch size of 4 samples resulting in 882 iterations (iter_{num}) per epoch.

The DNN achieves an accuracy of 96.5% on the validation set using 513×513 random crops like during training and 84.8% accuracy on the full resolution test set. A more detailed overview of the training and validation phase is listed in Table 2.

Dataset	Overall Acc.	Class Acc.	mIoU
Training	97%	80.2%	75.8%
Validation	96.5%	78.8%	74.3%
Testing	84.8%	67.1%	53.9%

Table 2: DeepLabv3+ (ResNet-101) performance overview on the train and validation set on the basis of 513×513 random crops, and the test set containing *full* resolution samples. The overall class accuracy indicates a non-biased class occurrence across the splits, otherwise the evaluation would denote a great performance drop.

[1] Experiments with different weather and lightning conditions are an easy extension of our framework and will be addressed in future work.

5.2. Results

To demonstrate the effectiveness of our approach, in this section we report on our experiments designed to investigate the relationship between domain-knowledge and detection quality of our SUT.

Correlation between domain-knowledge as instance-wise distance with IoU detection quality measure. As a first step, we investigate the correlation between distance and performance of our DNN. Figure 1 plots IoU, as our performance metrics of choice, over distance. The horizontal dotted line shows the average IoU over the test set if we do not consider the distance. In contrast, we see in the dark solid line the mean IoU at a given distance. [2] Illustrating the trendline by linregress from *scipy*, we can see a decreasing linear trend of IoU with increasing distance, *i.e.* at close distances the DNN performs considerably better than average, while for larger distances, *e.g.* $\approx 100m$, we would need to consider a performance drop. Hence, we can see that detailed performance evaluation is required for distance and needs to be studied closely for the considered task. Additionally, the plot visualizes the distribution of IoU with the (lower) 20% quantile and the upper 80% quantile, also showing the same trend. In the following, we further study this dependency of distance on IoU for individual samples.

Interpretation of the raw performance data from a verification-perspective. Figure 4 shows the raw intersection-over-union (IoU) evaluation of the semantic segmentation predictions for all pedestrians in the test data set. The visualized data is not averaged allowing us to take a *verification perspective* in contrast to a global *statistical perspective* [13]. By the term *verification perspective* we do not mean formal proof, but rather testing where each test must be passed. The basis of a verification perspective is a specification describing the required performance in detail. To this end, consider the following simple example of a concrete specification for the detection of a pedestrian given in Example 1.

Example 1 *A pedestrian is detected if at least one of its pixels is classified as the pedestrian class according to the semantic segmentation map.*

We can check this specification on the test set for every pedestrian, since we have a post-processing script for identifying the correspondence between pixels and pedestrian instances (see Section 4.2).[3] We evaluate such specifications

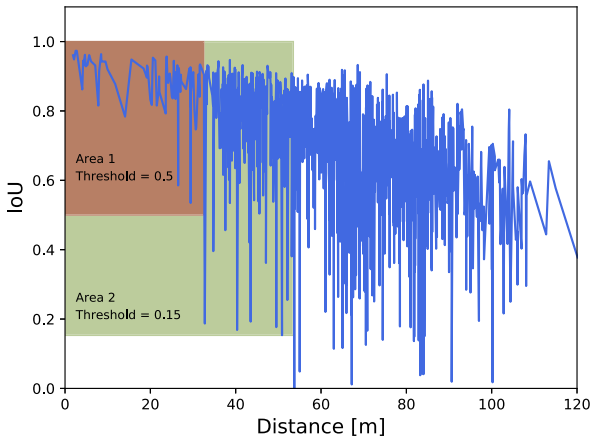


Figure 4: Unfiltered data of intersection over union (IoU) wrt. to distance of all pedestrian of the whole dataset.

with respect to known influence factors on performance. In the following, we use distance as an initial influence factor, which leads to the results shown in Figure 1. Based on these results, we can build a first, yet simple metric that couples distance and specification in terms of IoU. Using this interpretation, in Figure 4 we can see that up to the distance of approximately 50 meters not a single pedestrian was entirely missed in the test set, *i.e.* at least *one* pixel of the pedestrian were correctly classified as the target class. This view may be useful for portraying the semantic segmentation task as an object detector. To reduce sensitivity and thus the false positive rate, and the noise rate, we may use a threshold (δ) for the IoU that allows to control what percentage of pedestrian pixels needs to be classified correctly. To mitigate noisy detections for further experiments, *i.e.* single false positive pixels, constrains of a minimum pixel number n should be implemented. Hence, our follow up component necessitates bounding boxes containing at least n pixels raising the IoU's lower bound to *e.g.* 40%.

We define a metric called $dIoU_\delta$ as follows on the set of pedestrians $\mathcal{P}$, where each pedestrian $p \in \mathcal{P}$ has an associated distance $dist(p)$ and $IoU(p)$ (the blue line in Figure 4).

$$IoU_{dist}(d) = \min\{IoU(p) \mid dist(p) \leq d \wedge p \in \mathcal{P}\} \quad (2)$$

$$dIoU_\delta = \max\{d \mid IoU_{dist}(d) \geq \delta\} \quad (3)$$

The first line determines for a distance d the smallest IoU of any pedestrian up to that distance, *i.e.* a lower bound. The second line looks at a particular threshold δ and determines the maximum distance at which the lower bound of

[2] In the plot we aggregate windows of 50 samples and compute the mean as well as the quantiles. The trendline is computed using linregress from the scipy.stats Python package.

[3] Note that for test evaluation, the SUT does not need to be capable of distinguishing between pedestrian instances, this distinction is only needed for resolving the correspondence in the semantic (group) segmentation ground truth, but not for the prediction.

the pedestrian IoU is larger or equal. This maximum distance is our metric $dIoU_\delta$.

Let us look at the concrete example in Figure 4. We can see that using a threshold of $IoU \geq 0.15$, no pedestrian was missed within a distance of 54 meters, *i.e.*, $dIoU_{0.15} = 54m$ and using a threshold of $IoU \geq 0.5$, no pedestrian was missed within 33 meters, *i.e.*, $dIoU_{0.5} = 33m$.

By integrating the relevance aspect into the metric in Equation (3), *i.e.* here the distance, we can identify more precisely where the specification is fulfilled. This, in turn, allows to better understand whether the performance is acceptable for the task at hand. Of course, $dIoU_\delta$ is only an initial example of a relevance metric and more elaborate metrics will be needed. For a discussion of the importance of relevance metrics in a broader context, we refer to Abrecht et al. [1].

Remark: we found out that the outlier in Figure (4) near the distance of 52 meters resulted due to a problem of our data retrieval, as illustrated in Figure 5. We found that the perception function did not miss the pedestrian and that the performance of the function is actually superior than depicted by Figure (4). However, we decided to base this discussion on Figure (4), because outliers are common in computer vision functions and therefore this figure suits for illustration. From a verification perspective these are promising results, and these are made possible by combining the computer vision performance measure IoU with the task-oriented distance into a relevance metric.

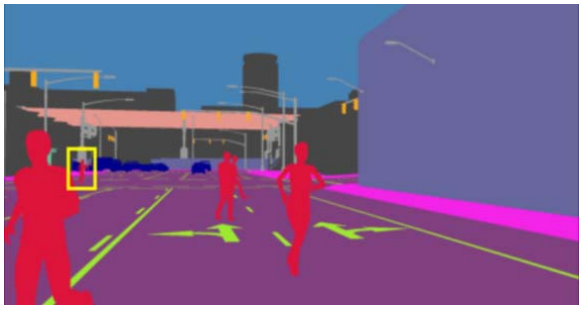


Figure 5: Downside of the forward projection from Algorithm 1 (see. Section 4.2) as a bounding box approximator.

Corner case scenario illustration. Figure 5 suits for visualization of a comprised outlier scenario showing the downside of our bounding box projection approach from Section 4.2. In this particular setting, due to an enlarged bounding box width, the projection captures two different pedestrians at distant locations from each other. Having a distance difference of approximately 52 meters and being part of the same bounding box, this inaccuracy results in an strong outlier (see Figure 4) with 2704 meters^2 variance. We assume that implementing instance segmentation for subsequent correlation studies, as connecting domain-knowledge with detection quality measures, it will decrease the spread by alleviating strong outliers enabling evaluation of crowded CARLA scenes.

6. Conclusions

In this paper we investigated the use of pedestrian distance as a proxy for their relevance in the context of a deep learning-based pedestrian detection. As a basis, we created a simulation setup based on CARLA for studying the effect of domain knowledge on performance ratings in controlled experiments. We used our setup for generating a dataset with pedestrians in different distances ranging from 2 to 120 meters. Our results on this dataset showed a linear decrease in performance for pedestrian detection in relation to their distance to the vehicle. Based on this information, we derived a first metric that characterizes a distance up to which all pedestrians are detected. Such metric provides more detailed information on detection performance that might eventually be used for arguing that the performance of a perception system is good enough for a specific task.

Since we faced scenario limitations due to unavailable instance segmentation in CARLA our future work aims to include an own implementation of that particular sensor, alleviating overlapping bounding boxes and mapping inaccuracies. Based on resulting distinguishable pedestrian pixels we strive to extend our relevance metrics using additional domain-knowledge as orientation, velocity, or more elaborate threat metrics like a Time-To-Collision (TTC) to refine task-oriented relevance metrics for verification of perception functions in automated driving.

References

[1] Stephanie Abrecht, Lydia Gauerhof, Christoph Gladisch, Konrad Groh, Christian Heinzemann, and Matthias Woehrle. Testing deep learning-based visual perception for automated driving. *Transactions on Cyber-Physical Systems*, 2021. to appear. 7

[2] Jan-Aike Bolte, Andreas Bar, Daniel Lipinski, and Tim Fingscheidt. Towards corner case detection for autonomous driving. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 438–445. IEEE, 2019. 2

[3] Eric Breck, Shanqing Cai, Eric Nielsen, Michael Salib, and D Sculley. The ml test score: A rubric for ml production readiness and technical debt reduction. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1123–1132. IEEE, 2017. 2

[4] Simon Burton, Lydia Gauerhof, and Christian Heinzemann. Making the case for safety of machine learning in highly automated driving. In Stefano Tonetta, Erwin Schoitsch, and Friedemann Bitsch, editors, *Computer Safety, Reliability, and Security : SAFECOMP 2017 Workshops*, volume 10489 of *Lecture Notes in Computer Science*, pages 5–16. Springer International Publishing, Cham, 2017. 1

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018. 2, 5

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685, 2016. 2

[7] John Dahl, Gabriel Rodrigues de Campos, Claes Olsson, and Jonas Fredriksson. Collision avoidance: A literature review on threat-assessment techniques. *IEEE Transactions on Intelligent Vehicles*, 4(1):101–113, March 2019. 2

[8] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*, 2020. 2

[9] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proc. Annual Conf. on Robot Learning*, pages 1–16, 2017. 2

[10] Sergio Fernandez and Joaquim Salvi. Planar-based camera-projector calibration. *Seventh International Symposium on Image and Signal Processing and Analysis (ISPA)*, 01 2011. 4

[11] Jelena Frtunikj. Practical experience report: Engineering safe deep neural networks for automated driving systems. In Alexander Romanovsky, Elena Troubitsyna, and Friedemann Bitsch, editors, *Computer Safety, Reliability, and Security*, pages 235–244, Cham, 2019. Springer International Publishing. 2

[12] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2D2: Audi Autonomous Driving Dataset. 2020. 2

[13] Christoph Gladisch, Christian Heinzemann, Martin Herrmann, and Matthias Woehrle. Leveraging combinatorial testing for safety-critical computer vision datasets. In *Workshop on Safe Artificial Intelligence for Automated Driving*, 2020. 6

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[15] Jonah Philion, Amlan Kar, and Sanja Fidler. Learning to evaluate perception models using planner-centric metrics, 2020. 2

[16] Timo Sämann, Peter Schlicht, and Fabian Hüger. Strategy to increase the safety of a dnn-based perception for had systems, 2020. 2

[17] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars, 2018. 2

[18] L. Spinello, A. Macho, R. Triebel, and R. Siegwart. Detecting pedestrians at very small scales. In *Proc. of the International Conference on Intelligent Robots and Systems (IROS)*, 2009. 2

[19] G. Volk, J. Gamerdinger, A. v. Bernuth, and O. Bringmann. A comprehensive safety metric to evaluate perception in autonomous systems. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, 2020. 2

[20] Oliver Willers, Sebastian Sudholt, Shervin Raafatnia, and Stephanie Abrecht. Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks. In *Computer Safety, Reliability, and Security. SAFECOMP Workshops*, pages 336–350. Springer, 2020. 1, 2

[21] Jianfeng Zhang. pytorch-deeplab-xception. https://github.com/jfzhang95/pytorch-deeplab-xception, 2018. 5